



O'Reilly Network | oreilly.com | Safari Bookshelf | Conferences

[Sign In/My Account](#) | [View Cart](#)

Articles | Weblogs | Newsletters | Meerkat | Learning Lab

Scale Software Agility with Rally

Agile Software Development
Management On Demand

Become a Certified ScrumMaster

Visit the ScrumMaster Portal

RALLY
software development
scaling software agility

advertisement

O'Reilly Emerging Technology Conference March 6-9 2006, San Diego, CA

Search

- Weblogs
- All of O'Reilly

[Most Recent](#) | [Webloggers](#) | [Topics](#) | [Top Weblogs](#)

Monthly Archives:

Sponsored by:

weblogs.oreilly.com: [Software Development](#)

- Login
- Register
- Manage Newsletters
- Register Your Books

Sites

- codezoo.com
- Databases
- Emerging Telephony
- LinuxDevCenter.com
- MacDevCenter.com
- Mozilla DevCenter
- ONJava.com
- ONLamp.com
- Apache
- BSD
- MySQL
- PHP
- Python
- Security
- OpenP2P.com
- Perl.com
- Policy DevCenter
- Ruby
- SysAdmin
- What Is...
- WindowsDevCenter.com
- Wireless DevCenter
- XML.com
- WebServices.XML.com

Affiliate Sites

- LinuxQuestions.org
- OSDir.com
- Servlets.com

Resource Centers

- Bioinformatics
- C/C++
- Databases
- Digital Media
- Enterprise Development
- Game Development
- Java
- Linux/Unix
- Macintosh/OS X
- .NET
- Open Source
- Oracle
- Perl
- Python

-Ofun



Geoff Broadwell
Oct. 05, 2005 02:05 PM
[Permalink](#)

- [Print](#)
- [Email weblog link](#)
- [Discuss](#)
- [Blog this](#)

URL: [http://www.pugscode.org/...](http://www.pugscode.org/)

Every project has a set of goals that guide it through the meandering path of development. For some projects, these goals are unspoken, seen only in the primary style of the code, or in the size and shape of its APIs. When [Autrijus Tang](#) started the [Pugs](#) project to create a Perl 6 compiler, he had an explicit goal: optimize for *fun*. Fondly referred to as `-Ofun` -- a typical compiler writer's joke, referring to the standard `-o` flag used to tell a compiler what its primary optimization goal should be -- optimizing for fun is probably the most important decision Autrijus made.

Optimizing for fun has had tremendous benefits. In just 8 months, the Pugs project has gained well over 100 committers, averaging about 30 commits a day for the life of the project. Unlike many projects, these commits do not all come from a handful of people. In fact, the 3 busiest developers can only claim about half the commits; the rest are well spread, with 50% of the developers able to claim 9 or more, 25% having 24 or more, and 10% having over 150 commits each!

The team is not just productive, it's also *creative*. Starting with just a single interpreted backend written in Haskell, Pugs has added compiled backends for JavaScript, Parrot, and Perl 5. Dozens of modules have been written or ported, ranging from encryption algorithms to IRC bots. Various developers have experimented with concepts ranging from continuations and coroutines to [self-referential preludes and efficient type inference](#), with working code often leading the official specs.

Of course, this should come as no surprise. As any cognitive science expert [will tell you](#), fun is a great way to focus the mind. Developers that aren't enjoying themselves will slow down, write buggy code, make poor decisions, and eventually leave the project (even one that pays). Conversely, rampant fun will bring coders in droves, and give them a passion for their work that shows in quality, quantity, and goodwill. It's a pretty good bet that optimizing for fun will produce a better product than almost any other method.

So what's Autrijus's secret for `-Ofun`? As he puts it, "the essence of fun boils down to instant gratification and a sense of wonder and discovery." Or as chromatic calls it, imagineering. It turns out there's quite a bit that goes into that:

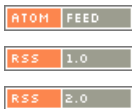
- Make `-Ofun` your primary goal (there can be only one). Next time you're forced to come up with a vision or mission statement, try that one on for size. (If management agrees, you've chosen a good place to work.) Every other goal chosen for the project should either flow from that one, or be secondary to it.
- Use modern, decentralized version control. If you're not already using a version control system, shame on you. If you are still using an old system such as CVS, RCS, or SourceSafe, you're really missing out. Modern systems offer atomic changesets (so all

Ads by Google

[Free JSP Editor - Eclipse](#)
M7 NitroX JSP, Struts, JSF IDE Professional Tools - Download now!
www.m7.com

[Perl Programmers](#)

[PYTHON](#)
[Scripting](#)
[Security](#)
[SysAdm/Networking](#)
[Web](#)
[Web Services](#)
[Windows](#)
[Wireless](#)
[XML](#)



Webloggers
[Login](#)
[Home](#)

edits relating to a single conceptual change can be captured together), full versioning of directories and symbolic links (so that files can be moved, copied, or renamed and still maintain full history), fast tags and branches, and more. Most important, modern version control systems offer decentralized, offline operation. Every developer can keep a local copy of the repository on their laptop, editing and committing locally to their heart's content, even when network access is unavailable. When ready for a merge, the developer can push changes to other developers or to a central "master" repository. Some systems, such as [darcs](#) and the [git](#) family are decentralized at their core; the excellent [SVK](#) client layers decentralization on top of a modern centralized system, [Subversion](#).

- Embrace anarchy. One of the key realizations of modern Internet projects (the oft-quoted [Web 2.0](#)) is that on the whole, your users can be trusted. The key is that the users also need to have the tools needed to repair any damage the tiny minority may cause. For a development project, modern version control systems can give you "anarchy with an audit trail". If something does go wrong (intentionally or more likely accidentally), it's easy for any other developer to identify and fix or revert the problem. Having this safety net allows the project to run full-bore without time-wasting process getting in the way, and without undue worry that code quality will suffer.
- Avoid deadlocks. There should be nothing blocking a programmer from committing his code. Mandatory reviews (or even acknowledgement) before commit are often used to work around failures in tools or project structure. For example, before atomic changesets and quality merge tools, it was extremely difficult to roll back a single change made at some point in the past; now it is much easier to do so. And without a proper test suite, it's hard to tell if a change broke the code in the first place. Any review process gets in the way of instant gratification (a key part of fun), and turns reviewers into bottlenecks - if they are too busy, the project may slow to a halt until they are free. Worse yet, a "bus error" (a key person being hit by a bus) may stall the project for good.
- Cast committer rights far and wide. A central core committer group is necessarily slower than allowing every developer to commit as desired. The more committers a project can gain, the faster the project can go, and the more ideas and group wisdom it can incorporate. Autrijus scans a number of technical groups 2-3 times a day trying to hand out the committer bit, responding to people's musings, and generally spreading awareness. It's important also to make committer sign-up fast and easy. Autrijus hacked a quick invitation interface into [rt.openfoundry.org](#) so that new committers could be invited en masse and sign up on their own without having to wait for an admin to fiddle with configs. This helps to make sure that people don't fall out of their interest window -- even the most casual contributors who just won't wait for any manual process. If invitation isn't completely automated (as for example with a wiki), make sure many people in different timezones have admin rights to invite a new committer, and pay attention enough to do so.
- Working code is more fun than mere ideas. Continuously push the team to sketch out ideas in code, committing quick and dirty prototypes that can be refactored as they grow. Have something to work with and show off to others from the first week of the project. Get things out in public, no matter what the state, as soon as possible. The easiest way to do this is to have publicly accessible version control. It should be trivial for someone to download the repository and play with it (and then easy later to edit and join in the fun). When his friends are afraid to release "not good enough" code, Autrijus often asks whether he can release it on their behalf, or says that he'd like to talk about their work, and can they please put it somewhere linkable? He does this so often that he's contracted it to a favorite one-word utterance: "url?"
- Build a rich, supportive community. There are numerous ways to do this, but most importantly, lead by example. Mentoring and even answering basic questions should happen continuously. Support groups should have quick turnaround (IRC is a good choice) and an open attitude. Encourage random fun tangents, such as the Perl community's JAPH, obfu, golf, and pervasive Tolkein poetry. Get every committer to add themselves to the AUTHORS file (this is a good choice to be a new developer's first commit, if they are unfamiliar with your version control system). Turn your project into a culture, one that you would like to live in.
- Excitement and learning are infectious. It's clear that everyone working on Pugs is having a blast, and the team is poring over technical papers, attending conferences, and trading information with other projects at a massive rate. There's a pervasive sense of high potential and great possibilities, and that sense decays slowly, even during inevitable lulls. All of this research and experimentation inevitably creates a ladder of skill, from wizened experts to fresh newbies. But that's actually a very good thing; skill ladders are part of the very [definition](#) of fun. True passion and community-building rarely develop around a project that doesn't have such a ladder. The more you know, the more you want to know -- and that's a heck of a lot of fun, cements the team, and produces some amazing code.

Many projects have achieved some of these by accident. Few have achieved all of them, and in such abundance. Autrijus gave us all a wonderful present when he made his decision to `-ofun` - now it's your turn.

[Geoff Broadwell](#) lives not far from O'Reilly headquarters in Santa Rosa, California, with a wonderful wife and daughter and four extremely spoiled cats. Geoff happily calls Perl the only

[needed](#)
 Join GetAFreelancer.com and bid on projects. Free and quick signup.
www.GetAFreelancer.com

[JOVIAL Software Migration](#)
 Automated translation tools/service Other custom tools and languages
<http://www.semanticdesigns.com>

[Perl Programming Jobs](#)
 Door to 200000+ Jobs Opens Here Apply Free - Join Now!
Naukri.com

[JOVIAL Compilers](#)
 20 Years of Safety-Critical Success Clients: BAE, Boeing, Lockheed etc
www.ddci.com

computer language he ever really loved, having sampled a fair number before and since. He is on a personal mission to prove that dynamic languages are by far the best programming option for almost every purpose, and believes that the ultimate Linux distro of the future will contain little more than a kernel, an OpenGL and X server, the Parrot VM, and many, many Perl scripts.

How do you -ofun?


You must be [logged in](#) to the O'Reilly Network to post a comment.

 Post Comment

 Main Thread Only

 Old First

Showing messages 1 through 4 of 4.


 **-of fun Asterisk->OpenPBX.org**
2005-10-11 11:26:58 smithnc [[Reply](#) | [View](#)]

A bunch of developers have forked Asterisk into OpenPBX.org and are enjoying the anarchy and freedom of fixing bugs and getting code in they could not get committed before. It's very infectious. and the enthusiasm keeps building. <http://www.OpenPBX.org>

 **-of fun Asterisk->OpenPBX.org**
2005-10-11 14:03:13 Geoff_Broadwell [[Reply](#) | [View](#)]

I think this is going to get relatively common unless the old guard starts opening up commits. A number of projects with tight committer rules have been forked of late -- it seems the benefits of free committership are starting to overwhelm the taboo against forking.

Personally, I think tight committer controls on OSS are exactly the kind of damage that one should expect get routed around.

 **Study confirms findings about fun**
2005-10-10 02:17:29 AHip [[Reply](#) | [View](#)]

With the study about "Fun and Software Development" I can confirm the findings of the -Ofun project (see http://www.foss.ethz.ch/people/lbenno/BLuthiger_Fun_SoftwareDevel_OSS2005.pdf).

 **Study confirms findings about fun**
2005-10-10 10:12:09 Geoff_Broadwell [[Reply](#) | [View](#)]

Thanks for the link!

Showing messages 1 through 4 of 4.

Return to weblogs.oreilly.com.

Weblog authors are solely responsible for the content and accuracy of their weblogs, including opinions they express, and O'Reilly Media, Inc., disclaims any and all liability for that content, its accuracy, and opinions it may contain.



This work is licensed under a [Creative Commons License](#).

Weblog authors are solely responsible for the content and accuracy of their weblogs, including opinions they express, and O'Reilly Media, Inc. disclaims any and all liability for that content, its accuracy, and opinions it may contain.

For problems or assistance with this site, email help@oreillynet.com